IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 16 May 2022, accepted 15 June 2022, date of publication 22 June 2022, date of current version 28 June 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3185244

Joint Optimization Via Deep Reinforcement Learning in Wireless Networked Controlled Systems

KANWAL ASHRAF[1], YANNICK LE MOULLEC[1], (Senior Member, IEEE), TAMAS PARDY[1,2], (Member, IEEE), AND TOOMAS RANG[1,2], (Senior Member, IEEE)

[1]Thomas Johann Seebeck Department of Electronics, Tallinn University of Technology, 19086 Tallinn, Estonia
[2]Department of Chemistry and Biotechnology, Tallinn University of Technology, 19086 Tallinn, Estonia

Corresponding author: Kanwal Ashraf (kanwal.ashraf@taltech.ee)

This work was supported in part by the Estonian Science Agency ETAg through the Center of Excellence 'EXCITE IT' under Grant PRG620, Grant PUT1435, and Grant TAR16013; in part by the "Tallinn University of Technology (TTU) Development Program 2016-2022" under Grant 2014-2020.4.01.16-0032; in part by the Cognitive Electronics (COEL) the European Union (EU) Horizon 2020 (H2020) under Grant 668995; and in part by the Context of the SUITED Project funded by the Parrot Franco-Estonian Hubert Curien Partnership.

ABSTRACT This paper proposes a deep Reinforcement Learning (RL) based co-design approach for joint-optimization of wireless networked control systems (WNCS) where the co-design approach can help achieve optimal control performance under network uncertainties, e.g. delay and variable throughput. Compared to traditional and modern control methods where the dynamics of the system are important for predicting a system's future response, a model-free approach can adapt to many applications of stochastic behaviour. Our work provides a comparison of how the control performance is affected by network uncertainties such as delays and bandwidth consumption under an unknown number of devices. The control data is transmitted under different network conditions where several applications transmit background traffic data using the same network. The problem contains several sub-optimization problems because the optimal number of devices is non-deterministic under network delay and channel capacity constraints. The proposed approach seeks to minimize control errors in wireless network control systems in order to improve Quality of Service and Quality of Control. This proposed approach is used and compared using three model-free RL Q-learning algorithms for high-throughput flow control in a double emulsion droplets formation application. The results show that the allowable number of devices for reliable network communication under bounded network constraints is 10 when using binary search. The control performance of the system without considering network effect in the reward function (Scenario 1) was good with the C51 algorithm; when including OMNet++ based network effect in the reward function (Scenario 2), the best performance was achieved with all three algorithms (C51, DQN, DDQN) with an exponential reward function, and only with C51 in the case of a linear reward function. Finally, under random network conditions (Scenario 3), C51 and DDQN performed well, but DQN did not converge. Comparisons with other machine learning and non-machine learning algorithms also highlight the superior performance of the utilized algorithms.

INDEX TERMS Wireless networked control systems, co-design strategies, reinforcement learning.

I. INTRODUCTION

The outset of this work in wireless networked control systems (WNCS) stems from a multidisciplinary research effort related to a microfluidics application. Microfluidics has enabled automation in the pharmaceutical and diagnostic fields thanks to the use of small reagent volumes, increased particle monodispersity with uniform drug composition, and efficient evaluation methods for drug testing [4], [49]. To integrate microfluidic devices in the consumer market, a highly synchronized flow rate is a major challenge to be addressed [64]. For example, in the case of liposomal drug delivery [46], [47] which is promising for high-throughput cell screening, double emulsion could help in the better

The associate editor coordinating the review of this manuscript and approving it for publication was Marco Anisetti.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

formation of droplets. The formation of double emulsion droplets depends upon the synchronized delivery of the reagents at a specific flow rate [56]. The formation of double emulsion requires at least four pumping units for generating emulsions and additional pumps for reagents delivery. To achieve a high flow rate, different techniques have been proposed, which include several microfluidic units working in parallel [34], [71]. This raises issues that include not only the control of the devices but also the data communication and storage for achieving efficient control in a high throughput production unit. Our previous research focused on integrating wireless Cyber-Physical System (CPS) concepts [8], [9] with bioanalytical devices, which could help with efficient control in a high throughput laboratory setup. Such a Cyber Bioanalytical Physical System (CBPS) integrates the physical and biological processes with the computation and communication domains, enabling an efficient remote operation of the processes, which is the future of laboratory automation.

In a CBPS, synchronization between the devices is important to ensure the overall stability and reliability of the system [73]. The fault tolerance and delay requirement restrictions put constraints on the overall performance of the system (as in the case of Ultra-Reliable Low Latency Communications (URLLC)) [18]. These factors are affected by delays introduced by the control systems, which include computation and prediction delays, as well as the uncertainties of the wireless networks, including queuing delays, transmission delays and backhaul delays [41], [43]. It is thus important to see the design of this sub-domain of CPS, i.e. WNCS, as a co-design problem, [12], [42] rather than an interactive design in which one design lies on top of the other. The control and information distribution aspects of the application can be exploited by looking at the co-design of WNCSs. The principle of co-design of networked control systems is well established [17] and Figure 1 shows the design framework for networked controlled systems (inspired by the above-mentioned work), which acted as a starting point for our work in WNCS. In this paper, we use reinforcement learning to compensate for the delays of the systems (both wireless and control) in order to optimize the overall system's error response reliability. The reason behind using model-free RL rather than model-based approaches is that when dealing with massive systems, the delay models are non-deterministic in nature; additionally, the physical dynamics of the system might be unknown.

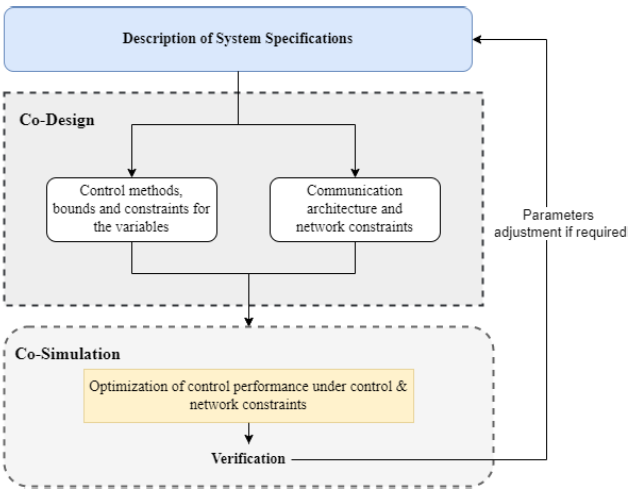

FIGURE 1. Framework for the co-design of networked controlled systems (inspired by [17]).

In addition, over a shared communication network, the traffic pattern [24] could be highly non-deterministic, specifically when dealing with event-triggered control; i.e., developing a traffic model over a shared communication network is also a non-deterministic problem. Because of factors such as data storage capacities and adaptability, simple search algorithms become infeasible when attempting to cope with changes in real-time as the problem complexity increases with network growth. The use of online learning algorithms could help solve these issues at the expense of convergence time as compared to offline algorithms, which require a lot of data. The proposed concept could even be extended to Ultra Reliable Low Latency Communication (URLLC) applications in which the system is subject to stringent delay constraints and system-wide optimization is necessary to achieve reliable performance.

A. SUMMARY OF CONTRIBUTIONS

Our contributions are summarized as follows:

1) We present our proposed joint optimization of WNCSs using a co-design approach. The aim is to analyze the benefit of using a model-free RL in stochastic systems as compared to classical and modern control methods.
2) To analyze the problem in-depth, classical optimization theory is used to formulate the problem. The objective of the problem is defined as the minimization of control errors under network constraints as well as errors introduced via the used reinforcement Q-learning technique.
3) The problem is extended for the application of droplet generation using a stepper motor where the flow rate is controlled by motor operation. To estimate the control delays as close as possible to reality, we performed the benchmarking of Raspberry Pi which is used as a central control unit of fluidic pumps in our laboratory setups. The wireless control of the pump is obtained via WiFi and the network uncertainties were mimicked using the OMNet++ simulation tool.

Furthermore, our proposed solution is evaluated under three different network scenarios:

Scenario1: The network uncertainties, such as delay and bandwidth consumption, were simulated using OMNet++. The optimal number of devices was calculated using binary search methods which satisfied the delay and bandwidth constraints for reliable performance. Finally, RL was performed using different algorithms, i.e., DQN, DDQN, C51,

and LSTM, and a comparison was made based on the convergence of the algorithms.

Scenario2: The delay and network data simulated via OMNET++ were used as a control factor during the RL environment design and were also used as a dynamic parameter in the reward function to obtain an efficient performance of the algorithm under the network uncertainties.

Scenario3: The network uncertainties were defined as random variables and were introduced in the RL reward function.

To mitigate the overestimation errors, a deep Q-learning algorithm is used instead of Q-learning and the performance of the methods is compared with other model-free RL algorithms including C51. Furthermore, our results show that double-DQN is more efficient to mitigate overestimation errors. These RL algorithms are equipped with an experience replay buffer [5] which acts as a middle ground between the offline and online algorithms; in turn this helps making convergence faster. An experience replay buffer is used to save the trajectory of previous experiences in order to improve the learning process's performance. The size of the previous observation must not be too small nor too large; i.e. updating the policy after each iteration will be extremely time consuming, and updating it after too many observations (which may overlook the pattern of change) will not improve performance.

B. DESCRIPTION OF THE APPLICATION

The formation of single or double emulsion droplets helps in high-throughput screening of the cell's susceptibility for drug formation and testing. There are several other applications of microfluidic droplets in the chemical industry [20] other than drug testing. In microfluidic applications, the formation of a double emulsion requires a synchronized flow of the different reagents. As mentioned earlier, the generation of such droplets requires at least four pumping units for emulsion generation, and more if needed. These pumping units require an efficient control method that guarantees the fluidic flow from each pumping unit at a specified flow rate.

Control of such pumps over wireless networks could add the possibility of cost-effective remote operation. However, if the systems are running in parallel with other high-data-consuming applications, such as video streaming, wireless communication may introduce additional challenges such as delay, packet loss, and channel congestion. Using classic control methods or robust control methods, e.g. Proportional Integral Derivative (PID) or Model Predictive Control (MPC) could be highly inefficient for applications with high synchronization requirements [28]. Indeed, one drawback of PID is that it is ineffective for Multi-Input Multi-Output (MIMO) systems and necessitates the tweaking of several parameters to get the desired response; one drawback of MPC it that it necessitates the modeling of the system's dynamics. The wireless network in a wireless control system is non-deterministic by nature, necessitating the continual adjustment of PID parameters or the development of the MPC model.

However, using a model-free (Black box) [57] or semi-supervised (grey box) implementation could help such a system achieve an optimal response. RL is based on trial and error methods and is derived from the field of psychology e.g. animal learning [45]; several pumping systems controlled over wireless networks will obtain the actions from the RL agents working in parallel and might learn from each other if necessary, as depicted in Figure 2. The use of the RL algorithms assists in adaptation of the system to a higher level without remodelling the system dynamics. Further details of RL and its comparison with other control methods are provided in the upcoming section II.

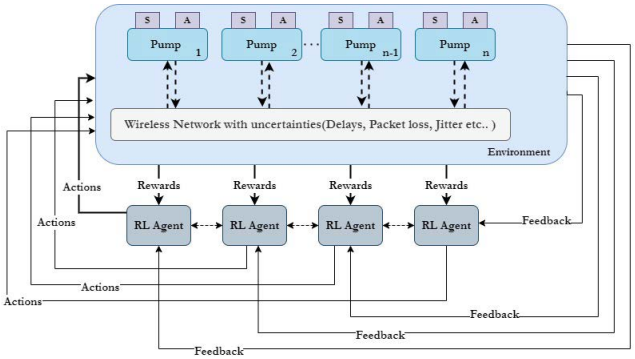

FIGURE 2. Reinforcement Learning Application for Distributed Systems.

C. PREVIOUS WORKS

Existing research in the topic of WNCS focuses on several elements of its design challenge such as stability, reliability, and energy efficiency. On the other hand, the use of deep learning is mostly studied in the case of URLLC [53]. In work [29], a hybrid approach which combines wireless connectivity with wired connectivity for control of Unmanned Aerial Vehicles (UAVs) has been provided. The proposed reliable VANET routing decision scheme is dependent on network conditions and is based on the Manhattan mobility model. In [38], a model-free deep RL based framework is analyzed for URLLC in downlink of OFDMA systems while optimizing the power. A delay sensitive joint optimization control studies has been carried out for networked control systems in [41] for multi-loop systems, emphasizing the importance of delay sensitivities in the design of optimal control and network policies. In [36], a clustering-based strategy for efficient energy optimization in embedded processors for wireless sensor networks is investigated in order to increase the lifetime of WSN nodes and enhance better utilization of resources. In the context of communication rivalry, an adaptive learning-based approach for vehicle-to-vehicle and vehicle-to-infrastructure communication has been presented in [52]. The gain settings of the PID controller are explored under the influence of non-linear delay using neural networks and ant colony optimization in study [63], but other critical network parameters such as packet error and channel capacity are not taken into

account. The use of RL algorithms has been examined in study [37] to handle the collision problem in vast IoT networks, with encouraging findings; the optimization problem is modelled as a function of access delay, access success and energy consumption rewards. A similar technique to ours has been investigated in [68] with the goal of optimizing platoon performance by accounting for wireless network delay and control stability. However, the work models the vehicle dynamics, which is a complex task in case the dynamics of the application are unknown or difficult to model. In [35], a framework for prediction and communication co-design has been provided for improving reliability of URLLC systems using optimization technique. However, a limitation of the mentioned work is that the implementation requires the information about the state transition of different parameters of the system. In [26], an offline scheduling algorithm has been proposed for machine-to-machine communications; however, as mentioned earlier offline algorithms are less adaptable to real-time changes. A joint optimization method for Quadratic Linear Regulator (LQR) cost and energy consumption is analyzed in [65], providing an energy-to-control efficiency framework for URLLC in IoT systems, but where factors such as channel capacity and number of users have not been taken into account.

II. REINFORCEMENT LEARNING VERSUS PREDICTIVE CONTROL

In reinforcement learning algorithm, an agent learns a strategy to control an environment based on feedback and reward strategy.

The state of the system is determined by valuation function Q(s, a) which is based on the sum of expected rewards R associated with previous states plus the discount factor γ related to next states. The overall reward R_t is given by:

$$R_t = r_t + r_{t+1}, \ldots, r_n \tag{1}$$

whereas the long-term reward is based on γ discount factor is given by:

$$R_{t+1} = r_{t+1} + \gamma R_t \tag{2}$$

For policy π that defines the probability distribution for any action a for state s, the valuation function is given by:

$$Q^{\pi}(s, a) = \mathbb{E}[\sum_{i=t}^{T} \gamma^{i-t} r(s_i, a_i)] \tag{3}$$

The valuation function tries to achieve an optimal value $Q^*(s, a)$ [58] where:

$$Q^*(s, a) = max_{\pi} Q^{\pi}(s, a) \tag{4}$$

Q-Learning is based on following Bellman update rule [6]:

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha(r_{t+1} + \gamma \max_{a_{t+1} \in A} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)) \tag{5}$$

where α denotes the learning rate. The reward function plays a significant role in RL; to reach a particular objective. The major challenges while designing a reward function includes positive infinitive loop in the feedback of a reward function as the objective would be achieved sooner while the agent has not still learned all the possible scenarios. Including a discount factor [39] in the reward function which comprises the factor affecting the overall performance of the system such as bandwidth assigned to each device after a certain device has left the network could help solve the infinite loop. The discount factor used in RL is similar to the quasi-hyperbolic discount as mentioned in Equation 6.

$$f(t) = \beta \rho^t \tag{6}$$

The quasi-hyperbolic discount function gets a value of ρ when $\beta = 1$; the discount factor solves the problem of positive loop in the infinite horizon as well as adds the contribution from the next states.

A. REINFORCEMENT LEARNING AND PREDICTIVE CONTROL

Predictive control of any system is regarded as an optimization problem where the problem is solved over a control horizon based on the system dynamics. Classic and robust control methods revolve around achieving a stable response of the system e.g. PID, LQR, MPC, etc. MPC has been in use for decades for solving networked control system problems thanks to its stable response [10], [67]. On the other hand, RL is based on agent(s) and an environment where the agent tries to learn the policy based on the feedback from the environment to solve an optimization problem through exploration and exploitation [66]. RL deals with how to learn control strategies by acting as an optimization framework for complex problems.

MPC algorithms might not converge in the real world where problems are more complex and non-deterministic in nature. Table 1 shows a brief comparison of RL with MPC and LQR. MPC might perform as close as to the RL algorithms for convex problems [15], but for WNCS where the network problem itself could be non-deterministic or non-convex in nature, MPC control will fail to solve the problem in an efficient manner [53] (see also Table 1).

B. MODEL-FREE REINFORCEMENT LEARNING ALGORITHMS

In RL, an agent learns the policy or valuation function based on dynamics of the system i.e. the model is given or learns the model of the environment with provided data or practical implementation [21], [27], [54]. RL algorithms can be model-based or model-free. In real-world problem where the system model might not be present or demonstration for a specific action is impossible, model-free algorithms could play an undeniable role. The model-free RL algorithms are divided into policy or valuation based learning techniques [33], [59] and are further classified into different algorithms as shown in Figure 3. In this work, our main focus was to highlight the use of value-based model-free algorithms

TABLE 1. Reinforcement learning vs control methods.

Methods	Complexity	Adaptability	Problem Convexity	Model Requirement	Robustness	Convergence Time	High-dimensional data handling
Reinforcement Learning (Model-based)	Low (Online), High (Offline) [16], [25]	High [32], [51]	Not required [32]	Yes [32]	Low [19], [32]	Low-to-Medium[1][16]	Good [40]
Reinforcement Learning (Model-free)	Low (Online), High (Offline) [16], [25]	High [32], [51]	Not required [32]	No [32]	Low [19], [32]	Low-to-Medium[1][16]	Good [40]
Model Predictive Control	High (Online), Low (Offline) [16], [25]	Low [32]	Required [32]	Yes [32]	High [32]	Medium-to-High[1][16]	Moderate [40]
LQR	Low [23]	Low [30]	Required[2] [48]	Yes	Moderate [30]	Low[3][44]	Moderate

[1] Depends upon dimensional complexity
[2] RL can help handling model-free cases
[3] Linear Convergence

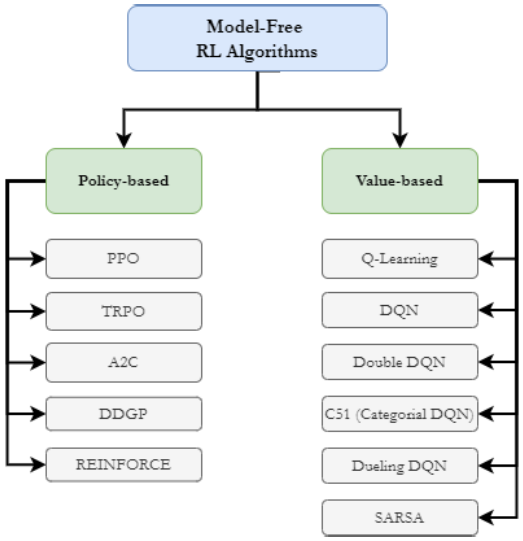

FIGURE 3. Model-free Reinforcement Learning Algorithms.

in wireless networked controlled systems. Although providing only the bounds or rules for the environment should be sufficient for these algorithms, we evaluated the response of the algorithms with deterministic data.

The algorithms chosen in this work are extensions of simple Q-learning algorithms including DQN, DDQN and C51 also known as categorical DQN. The only difference between Q-learning and DQN [62] is that the agent in DQN is based on neural networks rather than a simple Q-table. In DQN, an overestimation phenomenon is well observed due to the maximization function [7]. To solve this overestimation problem, DDQN uses two identical neural network models where one learns the Q-value and the other is a copy of the model learned from the last stage. Using a second model in combination with the current state model helps the system to evaluate different actions which might be more suitable for some states rather than the one on which the system is trained [1]. As compared to DQN or DDQN, categorical DQN uses a distribution value of the return rather than an expected value [11]. In multi-modal distributed data, where several peaks may be present in the data and a single average cannot truly represent the system's response, categorical DQN can solve the problem by looking at the distribution of the Q-function.

III. PROBLEM FORMULATION

The problem can be considered as a single task being completed by a number of centralized distributed event-triggered systems over a shared communication network where Table 2 provides definitions for necessary variables and symbols. Data from each system is timed stamped, un-synchronized and is transmitted under network imperfections (random delay, variable sampling time, packet drops, packet reordering). The tasks are divided into $p_1, p_2, \ldots, p_n$ systems and are controlled via a series of controllers $(c_1, c_2, \ldots c_n)$ with some or no inter dependency. The input of any single system will depend upon the learning parameter of controller i as well as

TABLE 2. Symbols & definitions.

Symbols	Definitions
R_t	Overall Reward
γ, ρ	Discount Factor, reliability parameter
$Q^*(s,a)$	Optimal Value of the Valuation function
$u_i(t)$	Controller Input
λ_i	Controller learning parameter
$y_1(t), \ldots, y_n(t)$	Controller outputs
$x_i(t+1)$	Controller state
$w(t), \zeta^a_{s'}$	Noise, External Noise
$\delta(\sigma)$	Trigger Coefficient
d_t, d_p, d_q, D_{max}	Transmission delay, processing delay, queuing delay, maximum allowable delay
$P_t, h, \sigma^2(B)$	transmission power, channel gain, noise spectral density
ξ^i_s	Synchronization parameter
$\mathbb{E}[\vartheta], e^s$	Upper bound on overestimation errors, synchronization errors
$\varepsilon_{max}, C_{max}, N_{max}$	Maximum allowable errors, maximum channel capacity, Maximum number of devices
K_{ij}	Communication Links

on the output of the same controller and on the output from other controllers.

$$u_i(t) = \lambda_i(y_1(t), y_2(t), \ldots, y_n(t)) \tag{7}$$

Here we are trying to minimize the delay and mean square error of the control system by prediction (model-free), where $u_i(t)$ is the input of the i^{th} system. Consider the i^{th} systems is defined by Equation 8 [50]:

$$x_i(t+1) = f(x_i(t), u_i(t), t) + w(t) \tag{8}$$

$$y_i(t) = g(x_i(t), u_i(t), t) \tag{9}$$

where $w(t)$ is the additive disturbance, $x_i(t)$ is the state of the system, $y_i(t)$ is the output of the i^{th} system and $u_i(t)$ is the input of the system. The control error of the system is given by Equation 10

$$e_c^i = y_r(t) - y_i(t) \tag{10}$$

The learning algorithm is designed to compensate for control errors and additive disturbances in order to follow the reference trajectory for optimum control performance as per Equation 11:

$$\lambda \Delta u^i(t) = e_c^i(t) + w(t) \tag{11}$$

A. DECISION VARIABLES | CONSTRAINTS

Several decision variables influence the control performance of the systems, including network and control constraints. The few variables that we included in our problem formulation are as follow:

1) TRANSMISSION ACK

The binary valued vector for transmission acknowledgment is defined such that:

$$\delta(\sigma) = \begin{cases} 1 & \text{Transmission happens at } \ t = \sigma \\ 0 & \text{No transmission is happening} \end{cases} \tag{12}$$

where δ is a function of σ(trigger condition)

2) DELAY CONSTRAINTS

The overall delay reduction for the system will ensure the stability of the system. In the case of a wireless networked controlled system with prediction and transmission happening over uncertain/un-reliable networks, the overall delay is the sum of transmission delay d_t, processing delay d_p and queuing delay d_q. The delay of the overall system is random in nature and could be modelled as Markov chains as if the network is under congestion so all the systems over the network will face delay. However, to ensure the stability of any system i, the system should satisfy the following constraint:

$$d_t^i + d_p^i + d_q^i \leq D_{max} \tag{13}$$

where the transmission delay is upper bounded by the maximum channel capacity and is given by:

$$d_t^i = \frac{N_p^i}{T_r} \tag{14}$$

where N_p^i are the bits to be transmitted and T_r is the transmission rate. A complete End-to-End delay model has been discussed in [43]. There exists an inverse relationship between transmission delay and effective bandwidth of network which eventually puts a bound on queuing delay. However, in the case of non-deterministic network, delay models (where an upper bound on the overall E2E delay and channel capacity is defined by separate tuning of different delay parameters) might not be required. For further details about the relationship between delay and number of devices one can refer to [55], [72].

3) CHANNEL CAPACITY CONSTRAINTS

To ensure the efficient utilization of resources and minimum transmission errors as well as packet loss, the information transferred by the cumulative systems should be less than the channel capacity. The channel capacity [13] is a function of bandwidth and Signal-to-Noise Ratio (SNR) and is given by Equation 15, where at any instance t the relation between channel capacity and bandwidth is given by:

$$C = B \ \ log_2(1 + SNR) \tag{15}$$

where the SNR can be represented as a function of transmission power P_t, channel gain h and noise spectral density σ^2.

$$C = B \ \ log_2(1 + \frac{P^t ||h||^2}{\sigma^2(B)}) \tag{16}$$

To ensure reliability of the overall system when N systems are transmitting, the upper bound on channel capacity is given by:

$$\sum_{i=1}^{N} B_i \ \ log_2(1 + \frac{P_i^t ||h_i||^2}{\sigma_i^2(B_i)}) \leq C_{max} \tag{17}$$

where N is number of devices and the upper bound on the number of devices (N_{max}) is effected by both capacity and delay constraints.

4) SYNCHRONIZATION ERRORS

The synchronization error [43] between i and j agents is given by:

$$e^s = \mathbb{E}[\sum_{j=1, i \neq j}^{N} K_{ij}(y_j(t) - y_i(t))] \tag{18}$$

where K_{ij} is communication links between the i_{th} and j_{th} agent. For simplicity, we define here a synchronization parameter ξ_s^i which depends upon how much output of agent i_{th} is delayed which will affect eventually output of j_{th} agent.

$$\xi_s^i = y_j(t) - y_i(t)) \tag{19}$$

5) OVERESTIMATION ERRORS

RL is based on learning optimal policies in the Markovian decision process where the objective function $Q(s, a)$ learns incrementally. The state learning depends upon reward r and discount factor γ as in Equation 21. In presence of

external noise $\zeta_{s'}^{a}$, the q-learning overestimation phenomenon occurs [61].

Lemma 1: Assuming the Q-learning happens under the stochastic environment with i.i.d variables $X = X_1, X_2, \ldots X_n$ which introduces a noise $\zeta_{s'}^{a}$ with zero mean in the evaluated function value, Q-learning overestimates in stochastic environments.

This phenomenon was first reported by Thrun, Anton in 1993. In presence of noises the evaluation function approximates as

$$Q_{t+1}^{approx}(s_t, a_t) = Q_{t+1}^{target}(s_t, a_t) + \zeta_{s_t}^{a_t} \tag{20}$$

where the target evaluation function is given by

$$Q_{t+1}^{target} = R_{t+1} + \gamma \max_{a_{t+1} \in A} Q(s_{t+1}, a_{t+1}) \tag{21}$$

The error introduced by the environmental noise in Equation 21 is given by:

$$\begin{aligned} \vartheta &= R_{t+1} + \gamma \max_{a_{t+1} \in A} Q^{approx}(s_{t+1}, a_{t+1}) - R_{t+1} \\ &\quad - \gamma \max_{a_{t+1} \in A} Q^{target}(s_{t+1}, a_{t+1}) \\ &= \gamma(Q^{approx}(s_{t+1}, a_{t+1}) - Q^{target}(s_{t+1}, a_{t+1})) \end{aligned} \tag{22}$$

The upper bound on this overestimation is given as in equation:

$$\mathbb{E}[\vartheta] \leq \gamma c; c = \epsilon \frac{n-1}{n+1} \tag{23}$$

The upper bound is well proved by Thrun, Anton and is included for the reader's convenience (Lemma 2).

Lemma 2: [60] While $f(x)$ denoting the density of noise variables $\zeta_{s'}^{a}$, in interval $[-\varepsilon, \varepsilon]$ i.e., $f(x) = Pr[\zeta_{s'}^{a} = x] = \frac{1}{2\varepsilon}$

B. OBJECTIVE FUNCTION

The goal is to maximize Quality of Service (QoS) and Quality of Control (QoC), which is achieved by minimizing synchronization and control errors, and is expressed as follows:

$$max(QoS \ \ and \ \ QoC)$$

Which is based on minimization of control errors e_c^i and noise $w(t)$ for i^{th} system.

$$minf(e_c^i, w(t))$$

The cost function for Mean-square control error is given by:

$$J_c^e = \mathbb{E}[\sum_{j=1, i \neq j}^{N} (y_r(t) - y_i(t))^T (y_r(t) - y_i(t))] \tag{24}$$

where $y_r(t)$ is the reference output. Based on constraints and optimization goal, the overall objective with the constraints is given as below:

$$\min(\mathbb{E}[\sum_{\substack{d_q, d_p, d_t \\ C_{max}, j=1, N, i \neq j}}^{N} (y_r(t) - y_i(t))^T (y_r(t) - y_i(t))]) \tag{25}$$

$$\text{s.t. } d_t + d_p + d_q \leq D_{max} \tag{25a}$$

$$\sum_{i=1}^{N} B_i \ log_2(1 + \frac{P_t}{B_i \sigma^2} z_t) \leq C_{max} \tag{25b}$$

$$N \leq N_{max} \tag{25c}$$

$$\delta(\sigma_i) \geq 0, \delta(\sigma_i) \in \{0, 1\} \tag{25d}$$

$$\mathbb{E}[\vartheta)] + e^s \leq \varepsilon_{max} \tag{25e}$$

$$\mathbb{E}[\vartheta] \leq \gamma c; c = \epsilon \frac{n-1}{n+1} \tag{25f}$$

$$e^s = \mathbb{E}[\sum_{j=1, i \neq j}^{N} K_{ij}(y_j(t) - y_i(t))] \tag{25g}$$

The objective is to reduce MSE under reliability constraints (25a, 25b, 25e) where constraint (25c) shows the upper limit on maximum number of devices and constraint (25d) is a feasibility constraint.

C. FORMAL DESCRIPTION

The importance of the use of formal methods in understanding the behaviour of stochastic systems has been discussed in our previous research [8]. Learning automata have been used for decades to solve complex problems like routing in stochastic environments [31]. In this context, RL provides the core of learning automata. A learning automaton based formal description of the problem could help to understand the considered problem in a perspective to replicate the approach for multi-agent systems as shown in Figure 4.

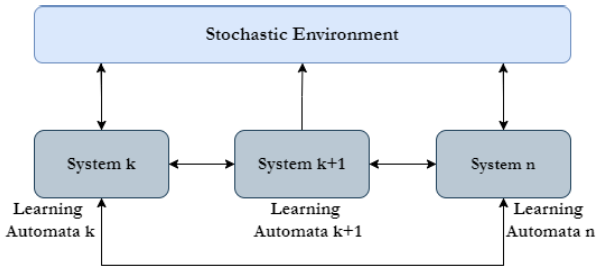

FIGURE 4. Learning Automata for mutli-agent systems.

This section provides the necessary definitions for learning automaton.

Definition: A learning automaton [3] is a tuple described as $L = (\eta, Act, P, p_t, u_n, \zeta_n, R)$

where:

$\eta \rightarrow$ Set of bounded input

Act $\rightarrow$ Defines the set of action in the action space $(a_1, a_2, \ldots, a_n)$

$\zeta_n \rightarrow$ Defines the sequence of environmental response. $\zeta_n \subseteq \eta$

$u_n \rightarrow$ Set of outputs/actions

P $\rightarrow$ Probability Space, which depends upon Probability and Sigma-Algebra function (F) of a set for bounded inputs and output sequence

$F_n = \sigma(\zeta_1, p_1, u_1; \ldots; \zeta_n, p_n, u_n)$

$p_t \rightarrow$ Set of probability distribution

$p_t = [p_n(1), p_n(2), \ldots, p_n(n)]^T$

$p_n(i)$ is conditional probability for the set of actions occurring under σ algebra function and sum of probabilities equates to 1

$p_n(i) = Pr\{\varsigma : u_n = u(i)|F_{n-1}\}$ where $F_{n-1} \subset F$

$R \rightarrow$ defines the reinforcement scheme where

$R_{t+1} = r_{t+1} + \gamma R_t$ where R_t represents the overall reward for the previous actions and γ is the discount factor

$\zeta_n \rightarrow$ conditional probability of the environment responses

$\zeta_t = [\zeta_n(1), \zeta_n(2), \ldots, \zeta_n(n)]^T$

IV. PROPOSED SOLUTION

As mentioned in the introduction section, a co-design approach offers a more satisfactory optimal control performance in the presence of wireless network constraints as compared to an interactive design approach. To formulate the problem, conventional optimization theory is used. The problem is formulated in mathematical form as indicated in Equation 25 with network constraints 25a, 25b, 25c, 25d, 25e, 25f and 25g. The problem under consideration is highly non-deterministic subjected that the number of devices (N) communicating is unknown. To solve the problem, the initial step is to calculate the maximum number of devices subject to channel capacity, delays and errors. Here we assumed that the minimum channel capacity required for each device to ensure delay and a small error probability is C^*. The constraint $(23g)$ comes into play for multi-agent interaction; to simplify the problem to a single agent, we have dropped the constraint from $(23g)$. Finding the solution to the problem consists of the following steps:

Learning: As mentioned earlier, to ensure optimal control performance a RL technique is used. Allocating a reward to the output response of the system in a stochastic environment under network constraints will help to achieve the desired performance. In case of error greater than the defined control threshold, the reward will be -1 i.e. a penalty, whereas in case of small error the reward will be $+1$. Section III-C provides a formal representation of the problem and the proposed algorithm 1 summarizes the approach used to solve the problem.

Reliability: To ensure reliability, the channel capacity constraint must be satisfied, which puts a limit on the maximum number of devices communicating. Thus, delays and errors are co-related with the assigned channel capacity. To make the problem simpler, constraints $23a$, $23b$, $23e$ are assumed to satisfy a reliability upper bound κ_{opt}. κ_{opt} provides minimum delay and errors under channel capacity constraints.

The maximum number of devices is obtained via a common binary search algorithm. Further discussion and explanation can be obtained from the simulation and results section V.

A. TIME COMPLEXITY ANALYSIS

1) TIME COMPLEXITY ANALYSIS FOR OUR APPROACH

For the proposed Algorithm 1, if a RL based approach is used, the computational complexity for *step*1 for determining the channel capacity for each user and *step*2 for the delay estimation for each user is $O(n)$. For *step*3, where the upper bound on the number of maximum allowable users is determined using a common binary search, the computational complexity is $O(logn)$. As for the while loop, the complexity is $O(n^2)$. The complexity of the value iteration algorithm is $O(S^2 \times A \times n)$ [22], where S are the states, A are the actions and n is the number of iterations. Therefore, the total time complexity of the proposed algorithm is given as

$$O(n + n + logn + (S^2 \times A \times n)^2)$$

Thus, the overall time complexity of the proposed algorithm becomes $O((S^2 \times A \times n)^2)$.

In what follows, we also present, for reference, the time complexity of approaches based on MPC and LQR.

Algorithm 1 Proposed Algorithm to Solve the Co-Design Problem

Require: End to End Delay, d^p, d^t, d^q, N, B_{max}, bit-rate, power, sensitivity, Signal-to-Noise Ratio(SNIR) threshold and trigger coefficient

Ensure: $\min(J_c^e)$

```
N ← n
Step1 :Determine Channel capacity for each user
Step2 :Determine Delay for each user
Step3 :Determine maximum number of allowable Users
while N ≤ N_max and C ≤ C_max do
    if y_ref − y_i is positive then
        r ⇐ r + 1
        y_ref ⇐ y_i                ▷ Dynamic Reference Change
    else if y_ref − y_i is negative then
        r ⇐ r − 1
    else if y_ref − y_i is zero then
        y_ref ⇐ y_i
    end if
end while
```

2) TIME COMPLEXITY FOR AN MPC-BASED APPROACH

Assuming that the model of the system is given, the relationship between the input and output variables of the system is known. The complexity of the algorithm remains the same as described above for our approach for *step*1, *step*2 and *step*3. However, for the while loop, the complexity for determining the output depends upon the number of inputs m and the prediction horizon p [70]; thus, the overall time complexity of an MPC-based approach under capacity and number of devices constraint is given as:

$$O(n + n + logn + ((m \times p \times n)^3)^2)$$

Hence, the overall time complexity will be $O(m \times p \times n)^3$ in case of conventional MPC and $O(m_i \times p \times n)^3$ in case of step-based MPC, where i represents the number of steps.

3) TIME COMPLEXITY FOR AN LQR-BASED SOLUTION

Assuming that the system dynamics are known and *step*1 − *step*3 remains the same, solving the control problem

(least-square) [14], [69] alone gives the time complexity as:

$$O(p \times n^3)$$

The total complexity, including the while loop, would turn out as:

$$O(n + n + logn + (p \times n^3)^2)$$

where p is the control horizon; thus the overall time complexity would be $O(p \times n^3)^2)$.

V. SIMULATIONS AND RESULTS

To evaluate the performance as close as possible to a real-life scenario, we obtained the network and control parameters for the pump used in our laboratory as shown in Figure 5. The pump is integrated with a Raspberry Pi (RPi) to implement a wireless controller over WiFi. The pump unit is a compact, portable, dual-channel piezoelectric pump that uses 2 Bartels mp6 piezo pumps in a closed-loop regulated pressure generator setup. The internal low-level controller is an ESP32 microcontroller, which will be connected to an RPi4 board. RPI4 benchmarking was performed to obtain an overview of its capabilities in terms of computation.

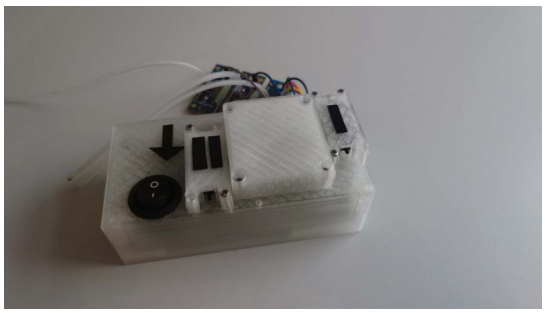

FIGURE 5. Our compact, portable, dual−channel piezoelectric pressure generator (i.e. pump) for droplet microfluidics application.

A. NETWORK SIMULATIONS

OMNET++ is a powerful C++ based simulation tool for wireless, wired and many other networks. OMNet++ was used to obtain network parameters such as delay and channel capacity for different numbers of devices. The results of the network simulations were aimed at acquiring End-to-End (E2E) delay, which consists of transmission delay (d_t), processing delay (d_b), propagation delay (d_p), and queuing delay (d_q) for control and background traffic applications. The network was simulated around 802.11e standards with the Quality of Service (QoS) service enabled and disabled [2]. In 802.11e, the MAC uses enhanced distributed channel access (EDCA) by which the video and audio packets sent can have different priorities, which helps achieve minimum delay in delay-sensitive applications. Using the same services, control commands were sent at the same priority level as video in 802.11e which enforces that control packets will be transmitted before the background traffic. The background traffic model represents unnecessary load over the network while transmitting control data. The network configuration included controllers with static processing delay defined as 5 *Sec*, server, configurator, Access Point (AP) and radio medium.

The upper bound on End-to-End (E2E) delay was defined as 200 ms for control applications. The bit rates were defined as 800 kbps and 33.3 Mbps for each control and background application, respectively. The maximum channel capacity was defined as 54 Mbps (2.4 GHz center frequency). The network simulations were performed for different numbers of devices i.e. 1, 5, 10,..., 15. Figure 6 gives an overview of the delay achieved for control devices versus background application when QoS is enabled for a single host. The control application experiences a constant and almost negligible delay whereas background applications experience a huge delay at the start and then tries to stabilizes; this initial delay is due to packet accumulation when the application initializes.

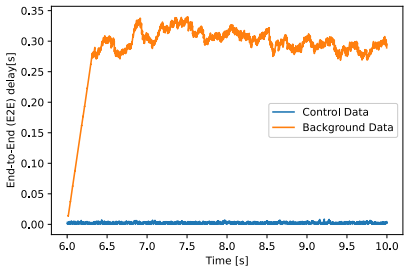

FIGURE 6. End-to-End Delay [s] vs. Time [s] when simultaneously transmitting control data and background data (QoS enabled).

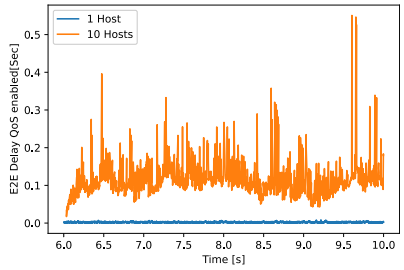

FIGURE 7. End-to-End Delay [s] vs. Time[s] when transmitting control data for 1 and 10 hosts (QoS enabled).

Figure 7 depicts how the delay increases when 10 hosts are communicating control data, versus the case with 1 host due to shared bandwidth. As the number of hosts increases, the delay experienced by the control applications also increases.

Next, when QoS is not enabled, the control applications are not prioritized and the control devices experience severe delay and low throughput. Figure 8 shows the throughput for the control versus background traffic when the QoS is not enabled.

The simulations were repeated for different numbers of host applications (N=1, 5, 10, 15); Figure 9 gives an overview of the maximum throughput achieved for control applications while QoS service is enabled.

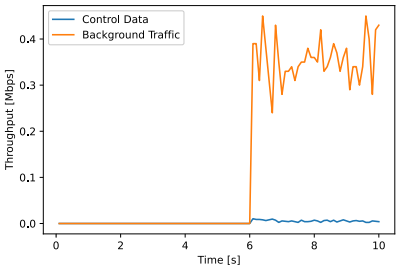

FIGURE 8. Throughput [Mbps] vs Time[s] when simultaneously transmitting control data and background data (Non-QoS).

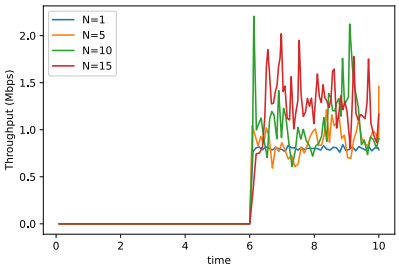

FIGURE 9. Throughput [Mbps] vs Time[s] when transmitting background data for 1, 5, 10, and 15 hosts (QoS enabled).

As compared with the non-QoS case, the throughput of the network changes over time, depending upon the data transmitted by high priority applications. Because of this, the throughput for control applications is comparatively higher than the throughput of the background applications. The average delay and throughput were calculated for control and background applications from the gathered data under QoS-enabled services; Table V-A summarizes the average delay and throughput achieved for a different number of applications.

Summary of the average delay and throughput achieved for different number of applications (hosts) with QoS enabled

Number of Hosts	Avg.Delay [ms]	Avg.Throughput [kbps]
1	2	32.4
5	102	39.2
10	119	52.6
15	245	65.6

B. REINFORCEMENT LEARNING RESULTS

As mentioned in the introduction section, the RL control of the pump was obtained under network uncertainties using three different approaches. To reduce the problem complexity, the agents were assumed to be performing independently from each other and the problem was solved for a single agent interacting with the environment where other agents are present and affecting the same environment. To add the effect of delay and bandwidth consumption parameter in the reward function, a reliability parameter ρ 6 was introduced in the reward function. For accommodating different possibilities where either delay or bandwidth consumed by the application exceeds the upper bound, which in turn leads to packet loss, the reliability parameter ρ was assigned a probability value between 0 and 1.

The ρ factor was introduced in two different ways: as a linear multiplier, as well as an exponential multiplier, to analyze the effect of it in the convergence of the algorithm.

Scenario 1:

In the first scenario, the network effects were not included in the reward function of the RL environment. Three algorithms, namely DQN, DDQN and C51 were used for the learning of the agent. In addition to the state of the system, the difference between allowable upper and lower bound of the flow rate was factored in the reward function. Figures 10, 11, and 12 show the average returns and loss for DDQN, DQN and C51 agents, respectively.

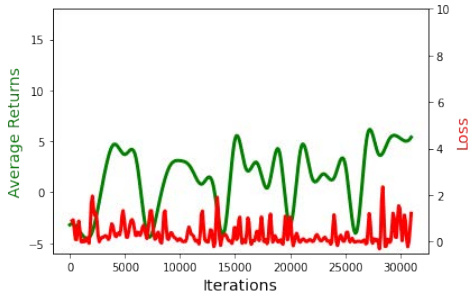

FIGURE 10. Average returns and loss for DDQN without network.

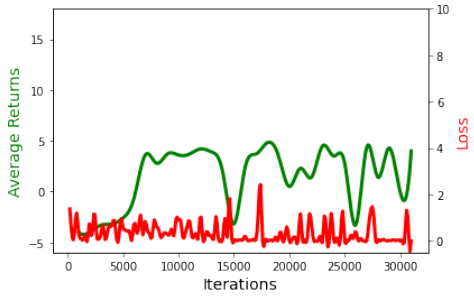

FIGURE 11. Average returns and loss for DQN without network.

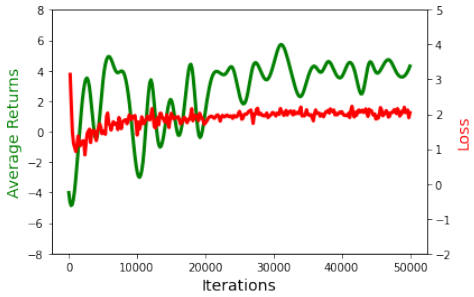

FIGURE 12. Average returns and loss for C51 without network.

The performance was best with the C51 algorithm where average returns were more stable; on the other hand, the performance with DDQN and DQN was poor.

Scenario 2: As mentioned earlier, to mimic the real network scenario, OMNET++ simulations were performed. In Scenario 2, the simulation results were included as a learning factor in the reward function either as an exponential or a linear multiplier.

The results with C51 algorithm with either exponential and linear rewards, see Fig. 13, outperformed DDQN and DQN with linear rewards.

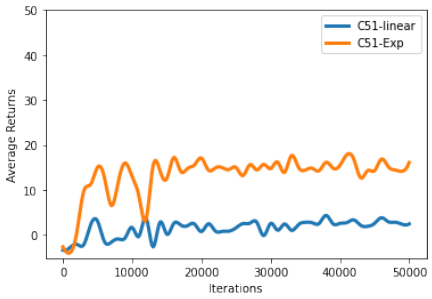

FIGURE 13. Average returns for C51 with network simulations with Linear and Exponential Rewards.

The results with DQN (see Fig.14) and DDQN (see Fig.15) with exponential reward performed well.

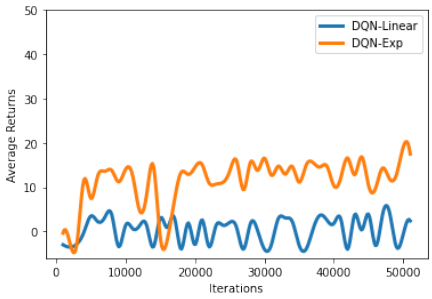

FIGURE 14. Average returns for DQN with network simulations with Linear and Exponential Rewards.

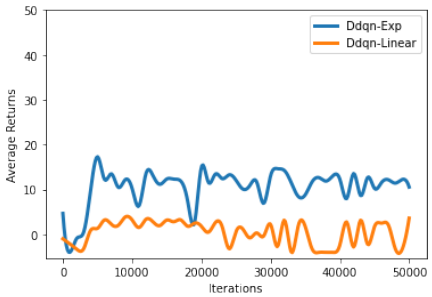

FIGURE 15. Average returns for DDQN with network simulations with Linear and Exponential Rewards.

Scenario 3: To accommodate more uncertain network scenarios, both bandwidth consumption and delay were introduced in the reward function as random variables.

Under random network conditions, C51 (see Fig.16) and DDQN (see Fig.18) performed well whereas DQN (see Fig.17) did not converged. This implies that although DQN performed satisfactory average reward when the problem was limited to a single agent but taking into account for network congestion or delay caused by other networks showed the unsuitability of the agent in high network traffic scenarios.

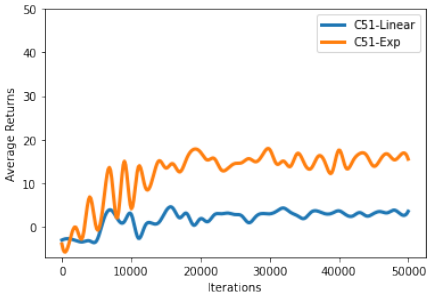

FIGURE 16. Average returns for C51 with random network conditions with Linear and Exponential Rewards.

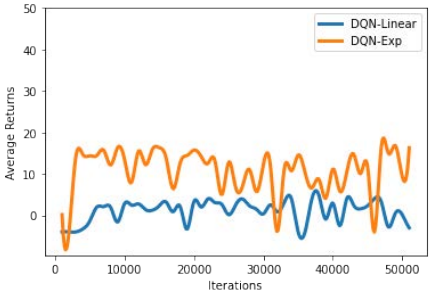

FIGURE 17. Average returns for DQN with random network conditions and Linear Rewards.

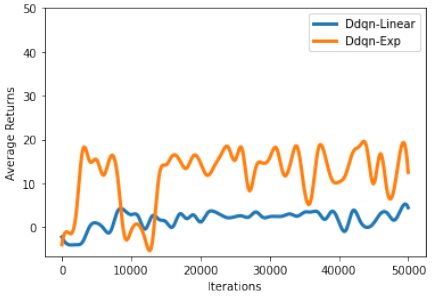

FIGURE 18. Average returns for DDQN with random network conditions and Linear Rewards.

Figure 19 shows the variation of flow rate obtained over 30000 iterations where network simulations were included in the reward function. It is evident from figure 19 that the DQN takes a bit longer to reach a stable response for

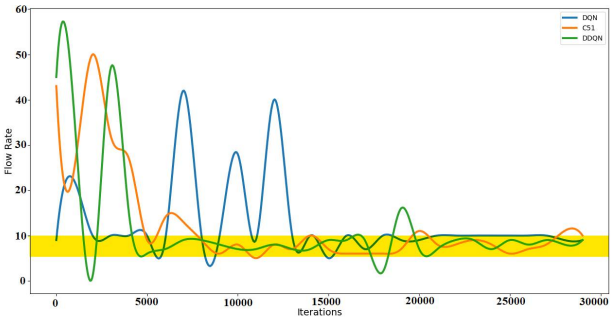

FIGURE 19. Flow Rate Prediction Under Network Constraints using OMNet++ Network Simulation data.

flow rate; however, DDQN and C51 reach stable response comparatively faster.

In addition to the above discussed scenario, we also analyzed other learning algorithms like LSTM using RNN for scenario 2. A reliability parameter depending upon E2E delay, number of devices and data rate was chosen between 0-1, the Poisson distribution of which was then used as a control error. A Poisson distribution for control error (based on network parameters) is assumed due to the non-deterministic nature of the system leading to no predefined distribution of control error. The total time of computation for 30,000 iterations was recorded as 4m 12 sec. Figure 20 shows the control error for prediction on training and test data for LSTM algorithm using an RNN dense layer. Both of them look back (previous timestamps) and the batch sizes were chosen to be 50, and 333 different events were used as input data. Covariance, Pearson's correlation, and Spearman's correlation between two test flow rate data inputs (lying within the constraints) and flow rate achieved by agents utilizing various RL algorithms were analyzed for further evaluation of the algorithms. Here, the covariance shows a linear relationship between the test data and the actions taken by the agent for achieving the optimal/desired response. The reported value in Table 3 is the covariance between the variables and itself; a positive value indicates a variable change in the same direction whereas a negative value suggests a change in the opposite direction. However, as the covariance is not a best measure to characterize the relationship between data

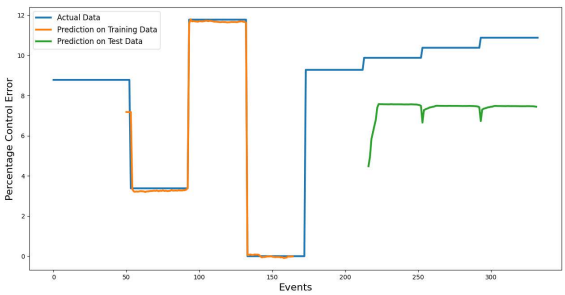

FIGURE 20. LSTM Control Error vs Events.

TABLE 3. Performance evaluation of reinforcement learning algorithms results.

Algorithms	Covariance	Pearson's Correlation	Spearman's Correlation
Test Data Input([10,6,6,10,6,10,6,6,6,6,10,6,6,6,10,6,6,6,6,6,6,6,5,6,6,6,8,6,6,6])			
DQN (Scenario1)	0.39	0.05	0.076
DQN (Scenario2)	-3.69	-0.19	-0.17
DQN (Scenario3)	0.12	0.008	-0.03
DDQN (Scenario1)	0.59	0.02	0.07
DDQN (Scenario2)	-0.57	-0.03	0.43
DDQN (Scenario3)	1.80	0.1	0.04
C51 (Scenario2)	8.49	0.61	0.45
C51 (Scenario1)	1.73	0.26	-0.07
C51 (Scenario3)	-0.4	-0.91	-0.09
Test Data Input ([10,6,6,10,6,10,6,6,8,6,10,6,6,6,10,6,6,6,8,9,10,10,8,10,6,10,8,6,10,6])			
DQN (Scenario1)	-0.22	-0.02	0.19
DQN (Scenario2)	-6.5	-0.29	0.03
DQN (Scenario3)	1.47	0.08	0.04
DDQN (Scenario1)	3.07	0.11	0.20
DDQN (Scenario2)	1.32	0.06	0.20
DDQN (Scenario3)	4.38	0.21	0.20
C51 (Scenario1)	-0.8	-0.61	-0.11
C51 (Scenario2)	6.04	0.36	0.39
C51 (Scenario3)	0.11	0.04	0.04
LSTM with RNN with OMNet++ Simulation Data			
LSTM (Prediction on Test Data)	0.031	0.22	-0.35

because it is hard to interpret, the Pearson and Spearsman's relationships between variables is also analyzed. The possible value of Pearson's correlation lies between -1 to 1, and values above 0.5 show a strong correlation between data in the same direction, while values below -0.5 show a strong correlation between data in the opposite direction. To account for a non-linear relationship between test data and agent actions, Spearsman's correlation was also calculated, where -1 shows a strong negative correlation and $+1$ shows a strong positive correlation. As evident from the results summarized in Table 3 for these evaluation metrics, C51 in scenario 1 and scenario 2 outperforms the other algorithms, whereas DDQN shows satisfactory results in some cases.

Another perspective to analyze is the role of the experience replay buffer. As mentioned earlier, C51 is an offline learning algorithm while DQN and DDQN are online learning algorithms; the experience replay buffer provides a middle ground for efficient operation. Based on the evaluation results, the role of the replay buffer in improving the overall performance of the C51 algorithm was further analyzed for Scenario 2. Different batch sizes were used to store the previous observations in the experience replay buffer and the cumulative rewards were calculated. It is evident from Figure 21 that increasing the batch size helps increase the average rewards in early stages of the learning process, and the use of a small batch size, leads to fewer cumulative rewards. However, drawing a conclusion that 'the bigger the batch size the better the performance' is not true as continuous observation and update improves policy. A frequent update of the observations is required, making sure that storing enough past history for the system to learn but not all of the observations. The overall simulation results showed that even if the random network

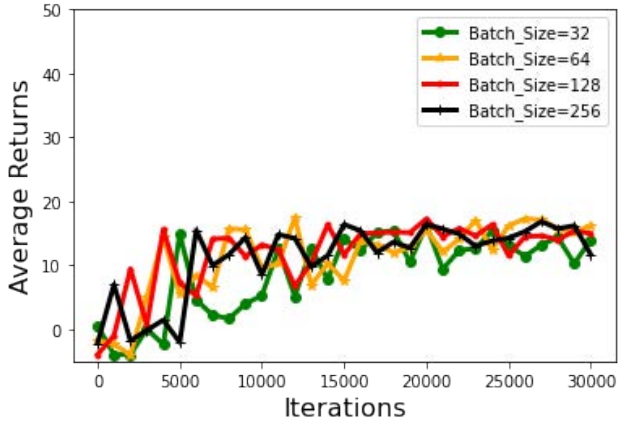

FIGURE 21. Replay Buffer: Batch Size vs. Average Returns for C51 Algorithm (Scenario 2).

conditions are used still C51 performs well to achieve an optimal control performance in a stochastic environment. Overall, the results show that reliability can be well achieved using model-free RL approaches. The scalability of the system is restricted by network constraints such as capacity and system requirements, i.e. delay; this sets an upper bound on the number of devices that can be supported under specific network conditions. Event-triggered network control reduces network load but introduces reliability issues which are out of the scope of this work. However, as the problem complicates, the method could take more time to converge. Although a binary search can provide an estimate for the optimal number of devices, it is best to include the network effects in the reward function for the system to learn the possible network scenarios for dynamic adaptation to network changes.

VI. CONCLUSION

In this work, we focused on the co-design for joint-optimization of wireless networked controlled systems using model-free RL. The research emphasized the importance of wireless network constraints in addition to the control system constraints to achieve an optimal system performance. The paper focused on many aspects of the problem in terms of optimization theory and argued the presence of various factors which motivated the use of RL as compared with classical and robust control methods. As a use case, the application of the theory was implemented for double emulsion droplet formation unit; DQN, DDQN and C51 algorithms were used to achieve the control performance of the system under bounded constraints. C51 was found to outperform the other algorithms due to its multi-modal problem-solving capabilities. The results also showed that the reward function plays an important role in the agent's learning process and that designing the reward function carefully could help to achieve better performance. Currently, our work does not investigate the reliability issues introduced via event triggered control for better efficiency. In the future, the aim is to explore a middle ground between better performance and reliability under different network scenarios using hybrid control approaches. Also, the power constraints have not been studied and are left for future work.

APPENDIX. LEMMAS

Lemma 1: Assuming the Q-learning happens under the stochastic environment with i.i.d variables $X = X_1, X_2, \ldots X_n$ which introduces a noise $\zeta^a_{s'}$ with zero mean in the evaluated function value. Q-learning overestimates in stochastic environments:

With the noise introduced during learning the reward attached with the Q-function is given as below:

$$r(\hat{s}, a) = r(s, a) + \epsilon$$

From probability theory the expectation of any variable X_i is given by its distribution over the N samples and can be formulated as below:

$$\mathbb{E}[X_i] = \frac{1}{|N_i|} \sum_{x_i \in N} x_i$$

At each stage the reward will be higher than expected due to cumulative errors added at each stage. Even if function values are too small at any stage, due to maximum operator in q-learning the function will tend to select the maximum from the estimated distributions ψ_i.

$$\max_{i \in 1,2,\ldots,n} \mathbb{E}[X_i] = \max_{i \in 1,2,\ldots,n} \mathbb{E}[\psi_i]$$

At each stage the estimation is made from the maximized estimates of the previous stages which leads to:

$$\mathbb{E}[\max_{i \in 1,2,\ldots,n} \psi_i] \geq \max_{i \in 1,2,\ldots,n} \mathbb{E}[\psi_i]$$

Under the assumption of ψ_i has non-zero probability the q-learning will tend to overestimate.

Lemma 2: [60] While $f(x)$ denoting the density of noise variables $\zeta^a_{s'}$, in interval $[-\varepsilon, \varepsilon]$ i.e., $f(x) = Pr[\zeta^a_{s'} = x] = \frac{1}{2\varepsilon}$

$$\begin{aligned}
\mathbb{E}[\vartheta] &= \mathbb{E}[\gamma(Q^{approx}(s_{t+1}, a_{t+1}) - Q^{target}(s_{t+1}, a_{t+1}))] \\
&= \gamma \mathbb{E}[\max_{a_{t+1} \in A} \zeta^a_{s'}] \\
&= \gamma \int_{-\infty}^{\infty} x \;\; n \; f(x) \left(\int_{-\infty}^{\infty} f(z)dz \right)^{n-1} dx \\
&= \gamma \;\; n \int_{-\infty}^{\infty} x \frac{1}{2\varepsilon} (\frac{1}{2} + \frac{1}{2\varepsilon})^{n-1} \\
&= \gamma \;\; n \int_{0}^{1} (2\varepsilon y - \varepsilon) y^{n-1} dy \\
&= \gamma \;\; n \;\; \varepsilon \int_{0}^{1} 2y^n - y^{n-1} dy = \gamma \;\; n \;\; \varepsilon (\frac{2}{n+1} - \frac{1}{n}) \\
\mathbb{E}[\vartheta] &\leq \gamma c; c = \epsilon \frac{n-1}{n+1}
\end{aligned}$$

APPENDIX. OTHER RESULTS

A. COMPARISON TO OTHER NON-MACHINE LEARNING METHODS

To support our argument of using RL instead of simple search algorithms, we analyzed the problem in more depth. For the optimization problem Equation 25, we tried to compute the linear approximation of the control error under constraints based on the desired output and input data-set gathered using simulations. For the computation, we used the well-known "*Newton Raphson Method*". However, the method failed to converge within the first 50 iterations under the subset of the acquired data. The use of "*Least Square Minimization*" and "*Trust Region Constrained*" algorithms was also considered, but as discussed earlier, the problem is non-deterministic in nature, which did not lead us to any feasible implementation. We also tried using "*Binary Search*" to solve other constraints of the problem, but the algorithm did not find any solutions with the provided simulation data-set. The use of a "*Brute Force*" algorithm was also tested; however, using the whole dataset led our system to run out of memory, or under best conditions, the algorithm was not able to find any solution in a 4-5 hour period. This led to trying the use of a smaller subset of the data; however, the algorithm did not manage to find the optimal/desired solution.

B. COMPUTATION TIME

For the simulations, we used a Lenovo IdeaPad L340 Gaming Laptop equipped with an Intel (R) Core (TM) i5-9300H CPU @ 2.4 GHz. Table 4 shows the computation time for the three scenarios for the implemented RL algorithms.

TABLE 4. Computation time taken by reinforcement learning algorithms.

Algorithms	Computation Time (30000 iterations)
DQN (Scenario1)	6 m and 14 s
DQN (Scenario2)	6 m and 38.9 s
DQN (Scenario3)	6 m and 40 s
DDQN (Scenario1)	5 m and 51 s
DDQN (Scenario2)	5 m and 54 s
DDQN (Scenario3)	5 m and 52 s
C51 (Scenario1)	6 m and 14 s
C51 (Scenario2)	6 m and 7.6 s
C51 (Scenario3)	6 m and 21 s

ACKNOWLEDGMENT

The authors would like to thank Osama Mohamed Mostafa Elgarhy for his helpful comments.

REFERENCES

[1] P. H. Moghadam, *Deep Reinforcement Learning: DQN, Double DQN, Dueling DQN, Noisy DQN and DQN With Prioritized Experience Replay*, Jul. 2019.

[2] *IEEE 802.11 Quality of Service-INET 4.3.0 Documentation*, OpenSim Ltd, Hungary, 2005.

[3] K. Najim, E. Ikonen, and A.-K. Daoud, Eds., "Optimization techniques," in *Stochastic Processes*. Oxford, U.K.: Kogan Page Science, 2004, ch. 3, pp. 167–221.

[4] *Insights Advancements Microfluidics*, MDPI, Basel, Switzerland, Sep. 2017.

[5] N. Agarwal, S. Chaudhuri, P. Jain, D. Nagaraj, and P. Netrapalli, "Online target Q-learning with reverse experience replay: Efficiently finding the optimal policy for linear MDPs," 2021, *arXiv:2110.08440*.

[6] O. Anschel, N. Baram, and N. Shimkin, "Averaged-DQN: Variance reduction and stabilization for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, Nov. 2016, pp. 176–185.

[7] O. Anschel, N. Baram, and N. Shimkin, "Deep reinforcement learning with averaged target DQN," 2016, *arxiv abs/1611.01929*.

[8] K. Ashraf, Y. L. Moullec, T. Pardy, and T. Rang, "Model-based system architecture for event-triggered wireless control of bio-analytical devices," in *Proc. 24th Euromicro Conf. Digit. Syst. Design (DSD)*, Sep. 2021, pp. 465–471.

[9] K. Ashraf, T. Pardy, and Y. L. Moullec, "Decentralized distributed data structure for bioanalytical laboratory setups," Tech. Rep., Jun. 2021.

[10] A. Aswani, N. Master, J. Taneja, D. Culler, and C. Tomlin, "Reducing transient and steady state electricity consumption in HVAC using learning-based model-predictive control," *Proc. IEEE*, vol. 100, no. 1, pp. 240–253, Jan. 2012.

[11] G. M. Bellemare, W. Dabney, and R. Munos, "A distributional perspective on reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 449–458.

[12] H. Benítez-Pérez, J. L. Ortega-Arjona, P. E. Méndez-Monroy, E. Rubio-Acosta, and O. A. Esquivel-Flores, *Control Strategies and Co-Design of Networked Control Systems: Considering Time Delay Effects*, vol. 13. Cham, Switzerland: Springer, 2019, doi: 10.1007/978-3-319-97044-8_1.

[13] A. Bensky, "Introduction to information theory and coding," in *Short-Range Wireless Communication*, 3rd ed. Boston, MA, USA: Newnes, 2019, ch. 9, pp. 211–236.

[14] P. S. Boyd, "LQR problem: Background," Tech. Rep. EE363, 2008.

[15] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[16] S. Brandi, M. Fiorentini, and A. Capozzoli, "Comparison of online and offline deep reinforcement learning with model predictive control for thermal energy management," *Autom. Construct.*, vol. 135, Mar. 2022, Art. no. 104128.

[17] M. S. Branicky, V. Liberatore, and S. M. Phillips, "Networked control system co-simulation for co-design," in *Proc. Amer. Control Conf.*, vol. 4, 2003, pp. 3341–3346.

[18] B. Chang, "URLLC design for real-time control in wireless control systems," in *Proc. IEEE 5G World Forum (GWF)*, Jul. 2018, pp. 437–439.

[19] S. Chen and Y. Li, "An overview of robust reinforcement learning," in *Proc. IEEE Int. Conf. Netw., Sens. Control (ICNSC)*, Oct. 2020, pp. 1–6.

[20] W.-L. Chou, P.-Y. Lee, C.-L. Yang, W.-Y. Huang, and Y.-S. Lin, "Recent advances in applications of droplet microfluidics," *Micromachines*, vol. 6, no. 9, pp. 1249–1271, Sep. 2015.

[21] I. Clavera, J. Rothfuss, J. Schulman, Y. Fujita, T. Asfour, and P. Abbeel, "Model-based reinforcement learning via meta-policy optimization," 2018, *arXiv:1809.05214*.

[22] D. Klein and P. Abbeel, "SP14 CS188 lecture 16—Bayes nets," Tech. Rep. CS188, 2014. [Online]. Available: http://ai.berkeley.edu/lecture_slides.html

[23] S. Dean, H. Mania, N. Matni, B. Recht, and S. Tu, "On the sample complexity of the linear quadratic regulator," *Found. Comput. Math.*, vol. 20, no. 4, pp. 633–679, Aug. 2020.

[24] L. Deng, C.-C. Wang, M. Chen, and S. Zhao, "Timely wireless flows with arbitrary traffic patterns: Capacity region and scheduling algorithms," in *Proc. 35th Annu. IEEE Int. Conf. Comput. Commun.*, Apr. 2016, pp. 1–9.

[25] D. Ernst, M. Glavic, F. Capitanescu, and L. Wehenkel, "Model predictive control and reinforcement learning as two complementary frameworks," in *Proc. 13th IFAC Workshop Control Appl. Optim.*, Jan. 2006, pp. 122–127.

[26] B. Farayev and S. C. Ergen, "Towards ultra-reliable M2M communication: Scheduling policies in fading channels," in *Proc. 23rd Int. Conf. Telecommun. (ICT)*, May 2016, pp. 1–6.

[27] V. Feinberg, A. Wan, I. Stoica, M. I. Jordan, J. E. Gonzalez, and S. Levine, "Model-based value estimation for efficient model-free reinforcement learning," 2018, *arXiv:1803.00101*.

[28] H. P. F. Fitzek, E. Steinbach, A. G. J. Cabrera, V. Latzko, J. Zhang, Y. Lu, M. Sefunç, C. Scheunert, R. Schilling, A. Traßl, A. Villamil, N. Franchi, and P. Gerhard Fettweis, "Communications and control," in *Tactile Internet*, F. H. P. Fitzek, S.-C. Li, S. Speidel, T. Strufe, M. Simsek, and M. Reisslein, Eds. New York, NY, USA: Academic, 2021, ch. 11, pp. 249–275.
[29] H. Gao, C. Liu, Y. Li, and X. Yang, "V2VR: Reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 6, pp. 3533–3546, Jun. 2021.
[30] A. A. Ghaffar and T. Richardson, "Model reference adaptive control and LQR control for quadrotor with parametric uncertainties," *Int. J. Mech. Mechtron. Eng.*, vol. 9, no. 2, pp. 244–250, 2015.
[31] R. J. Gibbens, "Teletraffic theory," in *Telecommunications Engineer's Reference Book*, F. Mazda, Ed. London, U.K.: Butterworth, 1993, ch. 5, pp. 5-1–5-14.
[32] D. Görges, "Relations between model predictive control and reinforcement learning," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 4920–4928, Jul. 2017.
[33] A. Harvey, K. Laskey, and K.-C. Chang, "Machine learning applications for sensor tasking with non-linear filtering," *Sensors*, vol. 22, no. 6, p. 2229, 2022, doi: 10.3390/s22062229.
[34] D. M. Headen, J. R. García, and A. J. García, "Parallel droplet microfluidics for high throughput cell encapsulation and synthetic microgel generation," *Microsyst. Nanoeng.*, vol. 4, no. 1, pp. 1–9, Apr. 2018.
[35] Z. Hou, C. She, Y. Li, L. Zhuo, and B. Vucetic, "Prediction and communication co-design for ultra-reliable and low-latency communications," *IEEE Trans. Wireless Commun.*, vol. 19, no. 2, pp. 1196–1209, Feb. 2020.
[36] T. Jackulin, M. Ramya, and C. Subashini, "Energy optimization for WSN architecture and self test embedded processor," in *Proc. Int. Conf. Emerg. Trends Electr. Eng. Energy Manage. (ICETEEEM)*, Dec. 2012, pp. 253–256.
[37] N. Jiang, Y. Deng, A. Nallanathan, and J. Yuan, "A decoupled learning strategy for massive access optimization in cellular IoT networks," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 3, pp. 668–685, Mar. 2021.
[38] A. T. Z. Kasgari and W. Saad, "Model-free ultra reliable low latency communication (URLLC): A deep reinforcement learning framework," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.
[39] W. B. Knox and P. Stone, "Reinforcement learning from human reward: Discounting in episodic tasks," in *Proc. 21st IEEE Int. Symp. Robot Hum. Interact. Commun.*, Sep. 2012, pp. 878–885.
[40] Y. Lin, J. McPhee, and N. L. Azad, "Comparison of deep reinforcement learning and model predictive control for adaptive cruise control," *IEEE Trans. Intell. Vehicles*, vol. 6, no. 2, pp. 221–231, Jun. 2021.
[41] M. H. Mamduhi, D. Maity, S. Hirche, J. S. Baras, and K. H. Johansson, "Delay-sensitive joint optimal control and resource management in multi-loop networked control systems," *IEEE Trans. Control Netw. Syst.*, vol. 8, no. 3, pp. 1093–1106, Sep. 2021.
[42] P. Martí, J. Yépez, M. Velasco, R. Villà, and J. M. Fuertes, "Managing quality-of-control in network-based control systems by controller and message scheduling co-design," *IEEE Trans. Ind. Electron.*, vol. 51, no. 6, pp. 1159–1167, Nov. 2004.
[43] K. S. Mazumder, *Wireless Networking Based Control*. New York, NY, USA: Springer, 2011, doi: 10.1007/978-1-4419-7393-1.
[44] H. Mohammadi, M. Soltanolkotabi, and M. R. Jovanovic, "On the linear convergence of random search for discrete-time LQR," *IEEE Control Syst. Lett.*, vol. 5, no. 3, pp. 989–994, Jul. 2021.
[45] F. E. Morales and H. J. Zaragoza, "An introduction to reinforcement learning," in *Decision Theory Models for Applications in Artificial Intelligence: Concepts and Solutions*. Harrisburg, PA, USA: IGI Global, 2011, pp. 63–80.
[46] B. S. Pattni, V. V. Chupin, and V. P. Torchilin, "New developments in liposomal drug delivery," *Chem. Rev.*, vol. 115, no. 19, pp. 10938–10966, Oct. 2015.
[47] M. Rahman, S. Beg, A. Verma, F. Anwar, A. Samad, and V. Kumar, "Liposomal-based therapeutic carriers for vaccine and gene delivery," in *Nanotechnology-Based Approaches for Targeting and Delivery of Drugs and Genes*, V. Mishra, P. Kesharwani, M. C. Iqbal, M. Amin, and A. Iyer, Eds. New York, NY, USA: Academic, 2017, ch. 4, pp. 151–166.
[48] A. Scampicchio and G. Pillonetto, "A convex approach to robust LQR," in *Proc. 59th IEEE Conf. Decis. Control (CDC)*, Dec. 2020, pp. 3705–3710.
[49] O. Scheler, K. Makuch, P. R. Debski, M. Horka, A. Ruszczak, N. Pacocha, K. Sozański, O.-P. Smolander, W. Postek, and P. Garstecki, "Droplet-based digital antibiotic susceptibility screen reveals single-cell clonal heteroresistance in an isogenic bacterial population," *Sci. Rep.*, vol. 10, no. 1, p. 3282, Dec. 2020.
[50] M. G. Losada, *Contributions to Networked and Event-Triggered Control of Linear Systems María Guinaldo Losada*. Cham, Switzerland: Springer, 2013, doi: 10.1007/978-3-319-34081-4.
[51] S. Sendari, "Study on robustness and adaptability of genetic network programming with reinforcement learning for mobile robot," Tech. Rep., 2013. [Online]. Available: http://hdl.handle.net/2065/40066
[52] W. ShangGuan, B. Shi, B.-G. Cai, J. Wang, and Y. Zang, "Multiple V2V communication mode competition method in cooperative vehicle infrastructure system," in *Proc. IEEE 19th Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2016, pp. 1200–1205.
[53] C. She, R. Dong, Z. Gu, Z. Hou, Y. Li, W. Hardjawana, C. Yang, L. Song, and B. Vucetic, "Deep learning for ultra-reliable and low-latency communications in 6G networks," *IEEE Netw.*, vol. 34, no. 5, pp. 219–225, Oct. 2020.
[54] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis, "Mastering chess and shogi by self-play with a general reinforcement learning algorithm," Tech. Rep., 2017, doi: 10.48550/arXiv.1712.01815.
[55] D. Singh, *Average Network Delay and Queuing Theory Basics*. Packet Pushers. [Online]. Available: https://packetpushers.net/average-network-delay/Online/e-learning/blog-post
[56] A. Stucki, P. Jusková, N. Nuti, S. Schmitt, and S. P. Dittrich, "Synchronized reagent delivery in double emulsions for triggering chemical reactions and gene expression," *Small Methods*, vol. 5, no. 8, pp. 1–8, 2021.
[57] F. Stulp and O. Sigaud, "Policy improvement methods: Between black-box optimization and episodic reinforcement learning," Tech. Rep., Oct. 2012, p. 34. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00738463
[58] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.
[59] C. Szepesvári," *Algorithms for Reinforcement Learning*, vol. 4. Edmonton, AB, Canada: Morgan & Claypool Publishers, Jan. 2010.
[60] S. Thrun and A. Schwartz, "Issues in using function approximation for reinforcement learning," in *Proceedings of the Fourth Connectionist Models Summer School*. Hillsdale, NJ, USA: Lawrence Erlbaum Publisher, 1999.
[61] H. V. Hasselt, "Insights in reinforcement learning: Formal analysis and empirical evaluation of temporal-difference learning algorithms," Utrecht Univ., Utrecht, The Netherlands, Tech. Rep., Jan. 2011.
[62] H. V. Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 30, no. 1, 2015, doi: 10.1609/aaai.v30i1.10295.
[63] Q. Xu and J. Yang, "Wireless networked control systems with neural networks and ant colony optimization algorithm," in *Proc. 37th Chin. Control Conf. (CCC)*, Jul. 2018, pp. 164–169.
[64] S. Yadavali, H.-H. Jeong, D. Lee, and D. Issadore, "Silicon and glass very large scale microfluidic droplet integration for terascale generation of polymer microparticles," *Nature Commun.*, vol. 9, no. 1, p. 1222, Dec. 2018.
[65] H. Yang, K. Zhang, K. Zheng, and Y. Qian, "Leveraging linear quadratic regulator cost and energy consumption for ultrareliable and low-latency IoT control systems," *IEEE Internet Things J.*, vol. 7, no. 9, pp. 8356–8371, Sep. 2020.
[66] M. Yogeswaran and S. G. Ponnambalam, "Reinforcement learning: Exploration–exploitation dilemma in multi-agent foraging task," *Opsearch*, vol. 49, no. 3, pp. 223–236, Sep. 2012.
[67] J. Yu, C. Kuang, and Y. Xie, "Model predictive control of NCSs with quantization packet dropout and time delays," in *Proc. 8th Int. Conf. Intell. Hum.-Mac. Syst. Cybern. (IHMSC)*, Aug. 2016, vol. 1, no. 3, pp. 160–163.
[68] T. Zeng, O. Semiari, W. Saad, and M. Bennis, "Joint communication and control for wireless autonomous vehicular platoon systems," *IEEE Trans. Commun.*, vol. 67, no. 11, pp. 7907–7922, Nov. 2019.
[69] H.-Y. Zhang, J.-Z. Luo, and Y. Zhou, "Explicit symplectic-precise iteration algorithms for linear quadratic regulator and matrix differential Riccati equation," *IEEE Access*, vol. 9, pp. 105424–105438, 2021.
[70] L. Zhang and J. Wang, "A novel multi-step model predictive control for multi-input systems," *Asian J. Control*, vol. 17, no. 2, pp. 707–715, Mar. 2015.
[71] Q. Zhang, J. Chen, and H. Gai, "High-throughput-generating water-in-water droplet for monodisperse biocompatible particle synthesis," *J. Mater. Sci.*, vol. 54, no. 24, pp. 14905–14913, Dec. 2019.
[72] Y. J. Zhang, S. C. Liew, and D. R. Chen, "Delay analysis for wireless local area networks with multipacket reception under finite load," in *Proc. IEEE Global Telecommun. Conf.*, Nov. 2008, pp. 1–6.

[73] Q. Zhu and A. Sangiovanni-Vincentelli, "Codesign methodologies and tools for cyber–physical systems," *Proc. IEEE*, vol. 106, no. 9, pp. 1484–1500, Sep. 2018.

KANWAL ASHRAF received the bachelor's degree in electrical engineering from the University of the Punjab, Pakistan, and the joint-international master's degree in smart system integration from Heriot-Watt University, U.K., USN, Norway, and BME, Hungary, in 2020. She is currently pursuing the Ph.D. degree with the Tallinn University of Technology, Estonia. Her research interests include application of wireless communication in CPSs and co-design of wireless networked control systems for bioanalytical applications.

YANNICK LE MOULLEC (Senior Member, IEEE) received the M.Sc. degree in electrical engineering from the Université de Rennes I, Rennes, France, in 1999, and the Ph.D. degree in electrical engineering from the Université de Bretagne Sud, Lorient, France, in 2003. From 2003 to 2013, he was a Postdoctoral Professor, an Assistant Professor, and an Associate Professor with Aalborg University, Aalborg, Denmark. Then, he joined the Tallinn University of Technology, Tallinn, Estonia, as a Senior Researcher and then on a professorship of Cognitive Electronics. He has supervised 11 Ph.D. theses and nearly 60 M.Sc. thesis. He was a Co-PI for the H2020 COEL ERA-Chair Project. His research interests include HW/SW codesign, embedded systems, reconfigurable systems, and the IoT.

TAMAS PARDY (Member, IEEE) received the M.Sc. degree in info-bionics engineering from Peter Pazmany Catholic University, Budapest, Hungary, in 2014, and the Ph.D. degree in electronics and telecommunication from the Tallinn University of Technology, Tallinn, Estonia, in 2018. He is currently a Senior Researcher with the Tallinn University of Technology. He has supervised one Ph.D. thesis and three M.Sc. theses. He has authored or coauthored 20 scientific publications. His research interests include flow- and temperature-control of lab-on-a-chip devices.

TOOMAS RANG (Senior Member, IEEE) was born in Tallinn, Estonia, in November 1951. He received the graduate degree in electronics engineering from Tallinn Technical University (TTU), Estonia, in 1975, and the Ph.D. degree in electronics from the Hungarian Academy of Sciences, in 1981. From 1984 to 1985, he was a Postdoctoral Researcher with Saarland University, Germany. In 1996, he was elected as a Professor in electronics design with TTU. In 1998, he was elected as the Head of the Department of Electronics. Since 2020, he has been a Professor Emeritus at TTU. He has successfully supervised 15 Ph.D. thesis. He is the author of about 120 scientific publications. His research interests include the topics from wide bandgap-based semiconductor devices and metallization technologies, till the microfluidics-based lab-on-chip devices.